Behavioral and Performance Characteristics of IPsec/IKE in Large-Scale VPNs

Okhee Kim
Advanced Network Technologies Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, Md 20899, USA
email: okim@nist.gov

Doug Montgomery
Advanced Network Technologies Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, Md 20899, USA
email: dougm@nist.gov

ABSTRACT

Cryptographic network security services are essential for providing secure data communication over an insecure public network such as the Internet. Recently there has been tremendous growth in the requirements for, and use of, secure *virtual private networks (VPNs)* to interconnect enterprises with business partners, traveling staff, and remote office locations.

IPsec tunnels have become one of the most widely adopted means to build secure VPNs between sites and individual computers. To date, most IPsec VPNs are statically configured and are of moderate scale. To facilitate future, very large VPNs with potentially varied security policies and changing memberships, the industry must move to the use of dynamic key management protocols and policy management systems to ease the administrative burden associated with VPN instantiation and operation.

In this paper we examine the dynamic behavior and relative performance characteristics of large scale VPN environments based upon IPsec and IKE version 1 (version 2 of IKE is currently under development by IETF). We use detailed, packet level, simulation models to characterize the performance impact of varying: key management scenarios, security association (SA) policy and management parameters, cryptographic algorithms, and implementation options in IPsec/IKE suites. Our results highlight the significant performance impact of subtle IPsec/IKE implementation and policy decisions on the overall performance and behavior of TCP based applications in large scale VPNs.

KEY WORDS

Network Security, IP Security Protocols, Internet Key Exchange and Management, Virtual Private Networks, Performance Analysis, Large-scale Networks.

1 Introduction

Cryptographic network security services are essential for providing secure data communication over an insecure public network such as the Internet. Recently there has been tremendous growth in the requirements for, and use of, secure *virtual private networks (VPNs)* to interconnect enterprises with business partners, traveling staff, and remote office locations.

The *Internet Protocol Security (IPsec)* suite [1, 2] was designed in the IETF to provide network security services such as confidentiality, data origin authentication, data integrity, and anti-replay to protect datagrams in the Internet. IPsec tunnels have become one of the most widely adopted means to build secure VPNs between sites and individual computers. To date, most IPsec VPNs are statically configured and are of moderate scale. To facilitate future, very large VPNs with potentially varied security policies and changing memberships, the industry must move to the use of dynamic key management protocols and policy management systems to ease the administrative burden associated with VPN instantiation and operation.

To date, little focus have been given to thoroughly understanding the behavior and performance impact of the interacting IPsec/IKE suite of protocols in very large scale VPNs. Characterizing the performance dynamics of such scenarios can provide users valuable aid in the design and management of scalable IPsec VPNs.

We use detailed, packet level, simulation models [3] to characterize the performance impact of varying: key management scenarios, security association (SA) policy and management parameters, cryptographic algorithms, and implementation options in IPsec/IKE suites. Our results highlight the significant performance impact of subtle IPsec/IKE implementation and policy decisions on the overall performance and behavior of TCP based applications in large scale VPNs.

In this paper, we examine the dynamic behavior and relative performance characteristics of large scale VPN environments based upon IPsec and IKE version 1 (version 2 of IKE [4] is currently under development by IETF). Simulation results are presented to characterize the relative performance impact of dynamic key management under varying SA granularity (e.g., *per-site* vs *per-host* tunnel granularity), re-keying strategies, and cryptograhic algorithms (e.g., 3DES, AES, HMAC_SHA1). We also discuss results that characterize the relative performance impact of distinct IPsec services, network topologies, and implemen-

tation options. The remainder of the paper is organized as follows: Section 2 gives a brief description of the IPsec protocol suite and Section 3 briefly describes the security models. In Section 4, we discuss the detailed experimentations and methods designed to analyze the behavioral and relative performance characteristics of interacting security protocols. Section 5 gives the analysis and observations of the experiment results. Finally, Sections 6 gives our conclusions.

2 Overview of IPsec/IKE

The IPsec protocol suite [1, 2] is a set of protocols that provide network security services at the network layer for a gateway or host. IPsec security services are provided by two security protocols: the Authentication Header (AH) [5] and the Encapsulating Security Payload (ESP) [6]. The Internet Key Exchange Protocol (IKE) [2] provides dynamic cryptographic key exchange and management functions. These mechanisms offered by IPsec are independent of cryptograhic algorithms and network topologies, which provide the flexibility for the selection of algorithms for the different security requirements.

AH provides data integrity, data origin authentication and optional replay protection for IP datagrams. ESP can provide confidentiality, data integrity, data origin authentication, anti-replay, and limited traffic flow confidentiality. Both AH and ESP support two types of encapsulation modes: transport mode and tunnel mode. A security association (SA) is an agreed-upon security information necessary to provide one-way secure communication between two peers (e.g., cryptographic algorithms, security protocols, keys, traffic afforded by this SA, etc). Two SAs are required to support bi-directional communication between two peers (one in each direction). IPsec allows the user to control the granularity of SAs (e.g., host-based or network-based tunnels) depending on the local security policy. The Security Policy Database (SPD) specifies what services are to be offered to IP traffic and in what mode of behavior. The SPD must be consulted for every single IP packet to see if it allows bypass, discard, or requires secure communication.

IKE provides automated cryptographic key exchange and management mechanisms in IPsec. IKE is used to negotiate security associations for use with Phase 1 and for other services such as IPsec (i.e., ESP and AH) (e.g., Phase 2). IKE defines a two-phase exchange: Phase 1 establishes an IKE SA through which the phase 2 exchange creates IPsec SAs. Phase 1 can be accomplished either in Main mode or Aggressive mode. Main mode provides identity protection. Aggressive mode can be used with reduced round-trips when identity protection is not needed. Both modes use the Diffie-Hellman public key exchange for shared secret keys. Phase 2 is accomplished in Quick mode that is used to negotiate SAs for other security protocols such as AH and ESP. Once phase 1 is estabiiehed either party can initiate Quick Mode since the IKE SA is bi-directional. Phase 2 generates two SAs, one in each direction. A single IKE SA can be used to negotiate more than one IPsec SAs. When perfect forward secrecy is desired, phase 2 exchange can initiate a new Diffie-Hellman exchange for a new and fresh keying material.

3 Security Models and Cryptographic Processing Overhead

We have modeled a detailed security simulator [3] using a ssfnet framework [7].

Cryptogrpahic operations in security protocols can impact the behavior and performance of the overall system. We modeled their impact on protocol funcitons and performance by allowing cryptograhic processing delay. To model these functions of security protocols, the processing time taken by specific cryptographic algorithms along with digital signatures and Diffie-Hellman exchange is simulated. The cryptographic processing delay is computed as a function of the performance (bytes/sec or Kbytes/sec) of a specific cryptographic algorithm and the size of a packet.

Our study focuses on the relative performance cost of a security system imposed by specific cryptographic processing overhead so that we can characterize what impact the cryptographic processing overhead has on the dynamics of IPsec and TCP based applications in large-scale VPNs.

4 Experimental Environment

Network Topology

We use a network configuration of size N in a full mesh topology. Each network consists of a single security gateway (SG) and M hosts behind it. For simulation experiments described in this paper, we use a full mesh network topology of size 10, each of which contains a single SG and 5 hosts behind each gateway (connected with a LAN). Generally, a SG is assumed to be an edge router connected to Internet with bandwidth of 1.5 Mbps (for IPec-enabled external interfaces) and to the local hosts with a LAN 100 Mbps (for an internal interface with no IPsec). For all the experiments, three model network configurations are examined:

1. ***asymmhost***: a network configuration where half the hosts in each network are clients and the other half of the hosts are servers (in this paper, we call it a *asymmhost* network configuration). In this configuration, hosts numbered from 0 to (M/2) in a network are exclusively clients and hosts numbered from (M/2+1) to M are exclusively servers;

2. ***asymmnet***: a network configuration where half the network contain only hosts that are clients and the other half of the networks contain only hosts that are servers. Odd numbered networks contain hosts that are exclusively clients and even numbered networks that contains hosts that are exclusively servers; and

3. ***fullsymm***: a network configuration where every single host in all networks acts as both a client and a server.

The tunnel granularity represents how a security system controls data traffic flows. We use two types of granularities: *per-site* creates a single SA for all the hosts behind the gateway and *per-host* creates an SA for each host.

In all cases, connections are made between SGs that contain clients and servers. Thus, each network that contains clients is connected to every other network that contains servers whenever TCP requests are generated from the local hosts, under various network configurations.

Table 1 shows traffic load (i.e., TCP sessions and IKE/IPsec SAs) generated from the network models. However, in this paper, we only present the results from a *fullsymm* network configuration for analysis and characterization.

Traffic Models

In all configurations, at the start of the simulation, each FTP client sends a request to a randomly chosen server to transfer fixed-size data. The TCP client then continuously sends the next request a random time seconds (e.g., Exponential with mean 10) after the completion of the session for the duration of the simulation time (e.g., 28800 seconds). The first FTP file transfer request is sent 10 seconds after the simulation starts. The first request time is randomized within a delayed start window time (in all cases 10 seconds) to avoid synchronized connections.

A security gateway initiates the SA negotiation for both IKE and IPsec SAs as "needed" basis. In other words, all initial SA negotiations are triggered by user traffic.

Performance Measures

We analyze, for this paper, a set of perfomance metrics. Specifically, we examine SA establishment latency (IKE SA and IPsec SA independently), which is a measure of time taken to set-up an SA by the initiator. IKE SA latency is measured for the time taken from sending the first IKE message and to receiving the last (6th) message (requiring 3 round trips). IPsec SA latency (for an outgoing SA) is measured for the time taken from sending a SA request (by IPsec module) and to receiving the corresponding SADB message from the IKE module. IPsec SA latency may include IKE set-up time if no IKE SA is established. When a corresponding IKE SA is available, IPsec SA set-up requires 2 round trips, including a *Delete* message or user traffic with the new SA, as a default mode.

For the application layer, we examine throughput, session latency, the number of sessions that succeeded, and total number of TCP retransmissions.

Experiment Background

All experiments are conducted in tunnel mode and use a time-based policy. For the initial SA establishment, the initiator starts the Phase 1 negotiation followed by Phase 2. Any security gateway can initiate the re-keying negotiation, if required by the local security policy. For the phase 2 re-keying, only outbound SAs are allowed to re-key.

For all the experiments, Phase 1 and Phase 2 use the same cryptographic algorithms, if required, and perfect forward secrecy for Phase 2 is used. The packet size is assumed to be constant 1000 bytes. We use one-way link delay as 50ms and no network interface delay is assumed. The packets are dropped when no SAs found. The IKE default authentication mode is a pre-shared secret key. The default IKE SA lifetime is 2400 seconds and IPsec SA 800 seconds.

The performance of cryptographic algorithms used for the experiments were taken from [8]. We scaled the performance figures, shown below, based on a Pentinum II processor.

Algorithms	Encryption	Decryption
3DES_CBC	1481KB/s	1481KB/s
HMAC_SHA1	15384KB/s	15384KB/s
AES_CBC	12500KB/s	12500KB/s
DH group 2	0.1sec.	

5 Results and Analysis

In this section we present simulation results to characterize the relative performance impact of security policies, implementation options in IPsec/IKE suites, and dynamic SA establishment.

5.1 Impact of Security Policies

The security policy specifies what modes of processing is required to IP traffic. The policy defines the level of protection to the traffic, if it requires secure communication, which includes the type of security services (e.g., authentication and/or encryption), the type of cryptographic algorithm (e.g, AES or 3DES), and SA granularity (e.g, *per-host* vs. *per-site*). We explore what impact these policy decisions have on the overall performance of file transfer protocol (FTP).

The following scenarios were considered: 1) Bypass: No sequrity is required; 2) AH with HMAC_SHA1; 3) ESP with 3DES and HMAC_SHA1; 4) ESP with AES and HMAC_SHA1; and 5) ESP with AES and null authentication.

Effect of SA Granularity

We show, in Figure 1, what affect SA granularity has on the overall performance of FTP. As shown in Figure 1, the

	asymmnet				*asymmhost*				*fullsymm*			
	per-site		*per-host*		*per-site*		*per-host*		*per-site*		*per-host*	
	1KB	1MB	1KB	1MB	1KB	1MB	1KB	1MB	1KB	1MB	1KB	1MB
Appl.: # of sess	70228	51482	70333	51452	84572	62078	84833	61622	140771	103793	137900	101382
IKE:												
Init req	25	25	25	25	45	45	45	45	45	45	45	46
Rekey req	389	388	389	388	700	699	699	701	702	701	702	700
IPsec:												
Init req	25	25	625	625	47	48	540	540	46	46	1125	1126
Rekey req	1214	1215	30048	29933	2283	2345	26147	26097	2237	2288	54402	57238
noSA pkts	26	26	625	625	49	48	540	540	48	48	1131	1126

Table 1. TCP Sessions and IKE/IPsec SAs Created From Each Network Configuration using *ESP (AES_CBC+HMAC_SHA1)*

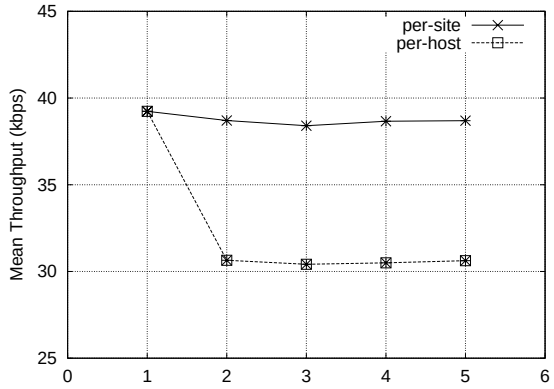

Figure 1. Effect of Granularity on the FTP Performance (file size of 1KB). X Label 1:Bypass; 2:AH(hmac_sha1); 3:ESP(3des+hmac_sha1); 4:ESP(aes+hmac_sha1); 5:ESP(aes+null)

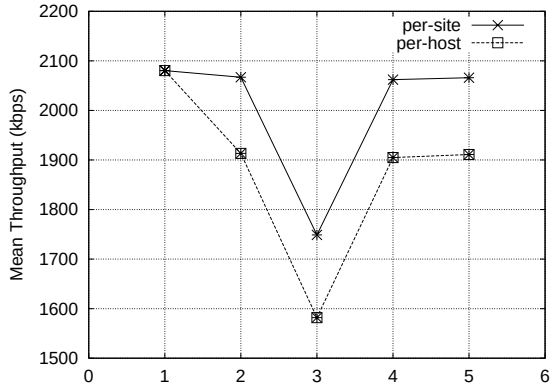

Figure 2. FTP Performance (file size of 1MB) using Various Cryptographic Algorithms. X Label 1:Bypass; 2:AH(hmac_sha1); 3:ESP(3des+hmac_sha1); 4: ESP(aes+hmac_sha1); 5:ESP(aes+null)

performance of FTP that uses *per-site* granularity is significantly higher than that of *per-host*. Since the number of SAs created is determined by the SA granularty, a *per-host* configuration creates more SAs, and consequently, requires more processing costs associated with them and needs increased wait time for traffic to wait for SAs being created, if needed.

Effect of Security Services

Figure 2 depicts the performance of FTP with vairous security services. As expected, the performance of ESP with AES is noticeably higher than that of ESP with 3DES. One point here is that, when used with a *site-based* SA configuration, the performance of all the cases improves greatly as high as 10%, in case of ESP with 3DES+HMAC_SHA1, due to the reasons described above. Table 2 shows the performance of SA establishments with regard to various cryptographic algorithms.

Note that latency for the IPsec initial establishment (requires 2 round-trips) is higher than that of IKE (needs 3 round-trips) for *per-site*, but is lower for *per-host*. This is because that all IPsec initial SA requests for *per-site* include the time for the IKE SA establishment since no corresponding IKE SAs are available at the time of the request. However, as for *per-host*, almost all IPsec SA requests only require the time necessarey to establish IPsec (e.g., 2 round-trips) since the corresponding IKE SAs are already established by the first IPsec SA request.

5.2 Impact of Implementation Options

Various Re-keying Techniques

We illustrate, in this experiments, the performance tradeoffs inherent in implementing alternate re-keying strategies.

The IPsec SA transfer techniques used in this experi-

	AH(h_sha1)		ESP(3des+h_sha1)		ESP(aes+h_sha1)		ESP(aes+null)	
	per-site	*per-host*	*per-site*	*per-host*	*per-site*	*per-host*	*per-site*	*per-host*
IKE:								
Init SA delay (s)	0.538	0.572	0.567	0.571	0.537	0.574	0.538	0.565
Rekey SA delay (s)	0.505	0.522	0.539	0.580	0.506	0.522	0.505	0.520
IPsec:								
Init SA delay (s)	0.847	0.331	0.894	0.344	0.842	0.331	0.850	0.331
rekey SA delay (s)	0.403	0.461	0.428	0.452	0.403	0.459	0.402	0.460

Table 2. IKE/IPsec SA Establishment Latency with Various Cryptographic Algorithms

ments are as follows:
1) *DeleteMsg*: set-up when a Delete message is received;
2) *fixed delay*: re-keyed SAs are set-up for use when either receiving inbound traffic with the new SA or a certain amount of time (e.g, 30s) has elapsed, in the absence of incoming user traffic;
3) *twice measured RTT*: re-keyed SAs are set-up for use when either receiving inbound traffic with the new SA or a twice the measured round trip time has elapsed, in the absence of incoming user traffic; and
4) *immediateUse*: SAs are used immediately. We include *immediateUse* only for the performance comparison.

As expected, *immediateUse* drops lots of packets due to no SAs available (as shown in Table 3). Almost all packets are dropped at the SA initiators because the corresponding SAs are expired. The SA initiators have already set-up the new SAs (and have consequently removed the old SA) as soon as Quick Mode 3rd message has been sent out whereas the SA responders still send data using the old SA until Quick Mode 3rd message is received. Even though there are lots of packets dropped at the security gateways, the server applications seem to manage to retransmit ACK segments due to no acknowledgements from the clients, thus, the application performance of *immediateUse* doesn't seem to be dramatically lower than those of other techniques in this experiment. However, the performance is likely to decrease when traffic load increases, resulting in serious security holes.

With the techniques used for SA transfer in this experiment, the amount of time to wait for inbound user data with the new SA can be one of factors that may affect the dynamic behavior (synchronization) and interoperability of the security protocols. As shown in Table 3, some of IPsec rekeying seem to have occurred during the off-time between the two consecutive FTP sessions. Some examples of such behaviors include changing the role of the initiator and responder during the rekeying procedures or a SA loophole in case of remote system crash. Thus, the SA transfer techniques seem to have a big impact on the dynamic behavior and interoperability with respect to IPsec re-keying due to the difficulties of SA synchronization between peers.

Handling Options of the First TCP SYN Packet

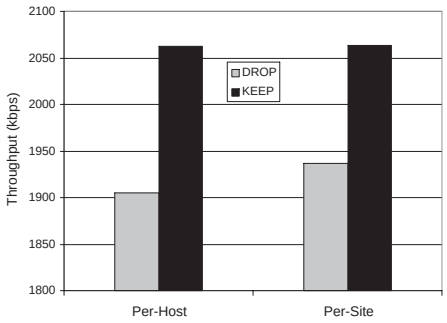

Figure 3. Impact of Handling Options on the Initial SYN Packets When No SAs Available. ESP with AES+HMAC_SHA1. File size of 1MB

We also investigate the impact of the handling of the first TCP packet (e.g., SYN message) (when no SAs are available) on the overall performance of TCP based applications.

The IPsec specification specifies that IP packets be dropped when no appropriate SAs found. However, we find that, as shown in Figure 3, the performance can be significantly improved when a SYN packet for a TCP connection is *kept* at the gateway, instead of dropping entirely, and transmitted to the peer as soon as the IPsec SA negotiation is complete, resulting in improving the performance (i.e., latency) as much as the initial retransmission timeout.

5.3 Impact of Dynamic SA Establishments

We explore the impact that the dynamic SA establishment has on the performance of TCP based applications. We use a simple dumbbell topology for this experiment in order to characterize the impact of dynamic establishment of different phases of the SA negotiation on the performance of an FTP. For the experiment, an FTP connection is applied with 3 scenarios: 1) dynamically created IKE SA followed by IPsec SAs; 2) statically created IKE SA and dynami-

	DeleteMsg	Fixed Delay	RTTD2	ImmediateUse
Application:				
# of sess (1MB/sess)	101382	99929	101419	101608
Avg Thrput (kbps)	1904.955	1900.938	1904.564	1896.463
Avg sess. delay (sec)	4.200	4.208	4.200	4.218
# of retransmissions	1126	1130	1145	2036
IPsec:				
Rekeying requests	57238	56297	57296	57249
rekeying SA delay (s)	0.459	27.755	0.918	0.309
pkts dropped (No SA)	1126	1129	1145	19839

Table 3. Performance trade-offs in Phase 2 Re-keying Techniques, file size of 1MB, ESP (AES_CBC+HMAC_SHA1)

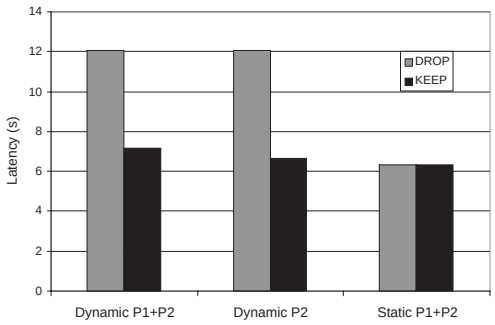

Figure 4. Impact of Dynamic SA negotiation on the FTP Performance (File size of 1MB)

cally created IPsec SAs; and 3) manually pre-installed IKE SA and IPsec SAs.

As depicted in Figure 4, the performance overhead for a dynamic SA set-up for a TCP based application increases, in latency in this case, as much as the initial retransmission timeout than that of pre-installed. The results show that the first packet of a TCP session (i.e., a SYN packet) is dropped due to no SAs available, resulting in a retransmission of the SYN by the application.

When a SYN packet is dropped due to no SAs, we also find that the performance of FTP makes no difference whether both IKE and IPsec SAs are created (case 1) or only IPsec SAs are established (case 2). This is because, in either case, an application has to retransmit the SYN packet since the gateway has dropped the packet.

However, we find that the performance can be significantly improved when SYN packets are *kept* at the gateway, instead of dropping entirely, as described in previous experiments. The performance of throughput (Figures not shown here) also show the identical characteristics as that of latency.

This indicates that the handlding of SYN packets when no SAs available can have a big impact on the performance of TCP based application.

6 Conclusions

In this paper, we have used security simulation models to examine and characterize dynamic behavior and relative performance of interacting suite of IPsec (with IKEv1) in large-scale VPN environments.

We have characterized the performance impact of dynamic key managements and security policies under various network configurations, and SA granularities. We have also highlighted the significant performance impact of dynamic SA establishment and IPsec/IKE implementaions on the overall performance and behavior of TCP based applications.

We believe that the relative performance characteristics and dynamics of an interacting suite of IPsec/IKE can provide users valuable aid in the design and management of scalable IPsec VPNs.

References

[1] S. Kent and R. Atkinson, *Security Architecture for the Internet Protocol*, RFC 2401, November 1998.

[2] D. Harkins and D. Carrel, *The Internet Key Exchange (IKE)*, RFC 2409, November 1998.

[3] http://www.antd.nist.gov/niist/

[4] C. Kaufman, Editor, *Internet Key Exchange (IKEv2) Protocol*, draft-ietf-ipsec-ikev2-11.txt, October 9, 2003

[5] S. Kent and R. Atkinson, *IP Authentication Header*, RFC 2402, November 1998.

[6] S. Kent and R. Atkinson, *IP Encapsulating Security Payload (ESP)*, RFC 2406, November 1998.

[7] http://www.ssfnet.org/homePage.html

[8] N. Ferguson and B. Schineier, Helix: Fast Encryption and Authentication, *Dr. Dobb's Journal, Computer Security*, #354 November, 2003, 28-34.